Dynamic Programming for Stochastic and Online Problems

Sports Strategy, “Secretary” Problems and Optimal Stopping

Joshua Knowles
School of Computer Science
The University of Manchester
In This Lecture

• Decision Making Under Uncertainty

• Optimal Play in Sports (including “Pig” Putting)

• The Secretary Problem — An Online Problem

• Optimal Stopping in a Parking Problem
Stochastic Dynamic Programming Problems

Dynamic programming problems are problems of sequential decision-making.

Each decision at each stage gives:
- a payoff
- a transition to a state at the next stage.

In a stochastic DP, either or both of the above can be random. But if we know the distribution of the random variables associated with the payoffs and transitions, then we can still find optimal policies.
Online Problems

So far in this course: optimization under full knowledge. No ‘hidden’ variables.

Life is usually more complicated. In business, in war, in games of skill, in sports, we have to make decisions based on partial information.

**Online Problems** in particular require decisions to be made (or generally actions to be done) as data arrives in a stream. Future data cannot be seen.

**Question:** given data seen so far, what is best possible course of action?
Dynamic Programming in Sports

If we know probabilities for achieving certain states, then we can optimize sports decision making.

**Darts**: given a number to finish, what is the best policy to follow?

**High jump**: when and by how much should the athlete raise the bar?
Estimating Probabilities On-the-Fly

It may be difficult to obtain all the probabilities needed to derive accurate optimal policies.

**Reinforcement Learning** methods estimate the probabilities (or the expected payoffs) at the same time as learning a policy.

Good for simulations/computer games/robotics/ autonomous systems. Not so good for humans to use for sporting strategy (calculations too laborious).

...Beyond the scope of our course.

Let us assume we know the probabilities already.
Decision-Making in Sports

Many sports test not only physical skill, but also decision-making. In dynamic programming, the set of decisions amount to what is called a “policy”. A policy states what to do in any given game state.

An example of a sport that involves decision-making is high jump. (Another is weight-lifting). In high jump, an athlete has only limited attempts at her disposal. She can pass at a height, but then she risks going to a level she can’t jump successfully. She needs to account for her own probabilities of clearing the bar at each height in order to make the optimal decisions of which heights to attempt.

Let’s look at another game involving decisions in more detail...
The Game, *Pig*

Pig is a competitive game for two players*. Each player initially has a **banked score** of zero. The aim is to reach a particular target **banked score** first. Player 1 goes first and makes *an attempt*. After each success, she can make another attempt, or “bank”. If she banks, her turn is over and the number of consecutive successes she had is added to her banked score. If she fails an attempt, her turn is over and nothing is added to her banked score.

The game continues in the same way for Player 2, and then alternates, until one player finally reaches the target banked score and WINS.

*It could also be played with more than two players.*
What policy is optimal for Pig?

Last year’s DP lab included a programming task to compute an optimal policy for Pig. The attempts in the game were putts (as in golf) from a certain distance.

Let’s play ...
What policy is optimal for Pig?

First of all we have to decide what is meant by an optimal policy.

For this game, the final score is just 1 (win) or 0 (loss). What we want to do is maximise the probability that we win. We need to know when to continue making attempts, and when it is better to stop and “bank”.

Let us look at an easy scenario.

The target score is 20. You are player 1. Your banked score is 18 and you have made one success. You must decide whether to bank or play. Your opponent is on a banked score of 19. What do you do?

Assume your success probability (per putt) is always 0.75.
Assume your opponent’s success probability (per putt) is 0.85. Then,

\[
p(\text{win}|\text{play}) = 0.75 + (0.25 \times 0.15) \times (0.75 \times 0.75 + 0.75 \times 0.25 \times 0.15 \times (\ldots) + 0.25 \times 0.
\]
\[
\approx 0.78
\]

\[
p(\text{win}|\text{bank}) = 0.15 \times (0.75 + 0.25 \times 0.15 \times 0.75 + \ldots)
\]
\[
\approx 0.12
\]

So, in this game state you would clearly do better to play than bank.

Now, assume you calculate these probabilities accurately, and \textit{memoize} them as follows

\[
p(\text{win})_{18,1,19} = 0.78.
\]

Then, one can compute the probabilities of winning from other game states too, recursively. For example the probability that I win from a
score of 17 vs 19 is the probability that I get the score to 18 vs 19 times the probability of winning from there.

If you want to try solving this (for fun!), then let me know and I will give you the lab question from last year ;-)
The “Secretary” Problem

You are to interview a series of people for a job. After each applicant has been interviewed you must decide whether or not to select them before interviewing the next applicant. When should you hire and stop interviewing?
The “Secretary” Problem

More details:

• We know $N$, the total number of applicants

• Each applicant has a ‘value’, which is drawn uniformly at random from the integers $\{1, \ldots, 10\}$. This value is ascertained precisely at the interview

• The objective is to maximize the expected value of the applicant hired

Note: This is just one version of a Secretary problem. (Another version states that the applicants can only be ranked relative to each other.)
The “Secretary” Problem

Transitions to next states are stochastic. Effectively we roll a die.

Payoff is stochastic. Although it is just the value of the current secretary if we decide to hire, this is random when we consider the previous decision before we know the value.

Recall: the expected value of a discrete random variable $v$ is $E(v) = \sum_v p(v) \cdot v$

(E.g. The expected value when we roll a die is 3.5)
The Dynamic Programming Solution

Let $f^*_n(X_n)$ represent the expected value of the reward under an optimal policy, at the $n$th stage, when the secretary just interviewed has value $X_n$.

\begin{align}
    f^*_n(X_n) &= X_n \quad \text{if you hire the secretary} \\
    f^*_n(X_n) &= E(f^*_{n+1}) \quad \text{if you continue}
\end{align}

Thus we have the recurrence:

\[ f^*_n(X_n) = \max(X_n, E(f^*_{n+1})) \]

with an extra stage $E(f^*_{N+1}) = 0$, meaning that we (expect to) receive no reward if we did not hire the Nth secretary.
The Dynamic Programming Solution

Let’s take $N = 3$. Then

$$f_3^*(X_3) = \max(X_3, 0)$$

We will hire the secretary, and hence our payoff is $X_3$. And

$$E(f_3^*) = \frac{1}{10} \sum_{X_3=1}^{10} X_3 = 5.5$$

When we look at the previous stage, we must solve:

$$f_2^*(X_2) = \max(X_2, E(f_3^*)) = \max(X_2, 5.5)$$

What decision does this imply making?
The Dynamic Programming Solution

If the second applicant scores 6 or above, then we should hire.

Working backwards further, we have:

\[ E(f_2^*) = \frac{5.5 \times 5}{10} + \frac{6}{10} + \ldots + \frac{10}{10} = 6.75 \]

and

\[ f_1^*(X_1) = \max(X_1, E(f_2^*)) = \max(X_1, 6.75). \]  

(3)

So choose the first applicant if and only if his value is 7 or above. Following this policy, the expected payoff overall will be 7.45.
Optimal Stopping

The policy for the secretary problem comes down to a decision on when to stop in order to achieve an optimal expected payoff.

A variety of other optimal stopping problems are known:

- When to stop dating to select a husband/wife
- When to stop advertising a good you are selling
- When to stop a gambling process
- When to stop bidding in an auction
- When to exercise an option (finance)
- When to stop searching for a job (similar to the secretary problem but from the applicant view).
Optimal Stopping for A Car Park Space

Driving past car park spaces, when should one take one in order to be closest to the shop entrance?

- Spaces can’t be seen until we get there
- Spaces occupied with some independent probability
- Cannot turn around (or would incur cost)
Summary

- Stochastic problems have random transitions or random payoffs or both

- If we know the probability distributions, we can still derive optimal policies using dynamic programming

- Optimal policies are often given in terms of expected payoff

- Could use other objective functions, such as maximizing worst-case payoffs.

See the webpages for links and more info.